Team Arsenault

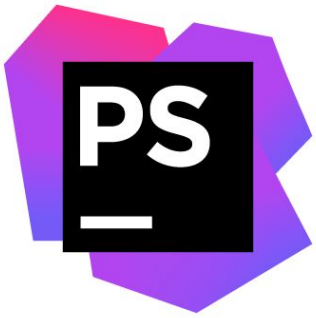
Technical Documentation

AS Capstone Summer 2016

Ian Arsenault & Jameson Arsenault

Website Tech Document

Website Development Tools:
Jetbrains Webstorm & PHPStorm



Github > Project Repository for collaborating on the project

CSS Frameworks: ***Materialize CSS***

Website File Structure:

```
Website
|-- C#
|   `-- application.exe
|-- Javascript
|   |-- materialize.min.js
|   `-- script.js
|-- Fonts
|   |-- raleway.eot
|   `-- roboto.eot
|-- CSS
|   |-- materialize.min.css
|   `-- styles.css
|-- Images
|   |-- favicon.ico
|   |-- banner.jpg
|   |-- app1.jpg
|   |-- app2.jpg
|   `-- app3.jpg
|-- Models
|   |-- dbConn.php
|   `-- functions.php
|-- Views
|   |-- landing.php
|   |-- main.php
|   |-- header.php
|   |-- footer.php
|   |-- checkout.php
|   `-- complete.php
|-- index.php
|-- sessionCheck.php
`-- protechdb.sql
```

Design Patterns

Will use two specific fonts (Raleway & Roboto) that compliment each other for headers and paragraph text.

Models folder will store dbConn.php and functions.php is where the backbone of the checkout system works. dbConn.php connects to the MySQL database, and functions.php will store all the functions to get and post information to the database

Views folder will store the viewable sections or templates used for to complete each page of the website.

Index.php will be the controller page for the entire website.

Commenting the code throughout will allow anyone to read and understand what is happening, as to help anyone who may jump in at some point in the future.

Design layout entails a landing page for the first time loading the website, animations, and hover effects.

Database:
MySQL

Table Names:
Customers - holds customer information such as name, address, email, etc.
Ccdata - holds customer id, and credit card information pertaining to that customer id
Orders - holds order information, first & last name, email of person who ordered

Relationships:
Customers(1) --> CCData(1)
Orders(1) -->Customers(1)
Customers(1) → Orders(Many)

Coding Conventions

The naming structure of the website will be as follows

CSS Styling

Classes -> .hover-active
Id's -> #nav-btns

PHP

Functions -> function new_Customer ()
Functions -> function get_UserId ()

PHP Variable -> $lastName
PHP Variable -> $orderNumber

MySQL

Table names:
customers -> all lowercase

Database Column Names:
fName -> camel case like php variables
ccExpYr -> each word following the first initial word or letter starts with a capital letter

C# Tech Document

Project Development Tools

C# program developed using:

Metro Modern UI Metro Framework v1.4.0.

Github > Project Repository

File Structure

```
PatientRecords
|-- PatientRecords
|   |-- bin
|   |   `-- Debug
|   |       |-- PatientRecords.vshot.exe
|   |       `-- PatientRecords.vshot.exe.manifest
|   |-- obj
|   |   `-- x86
|   |       `-- Debug
|   |           |-- TempPE
|   |           |   `-- Properties.Resources.Designer.cs.dll
|   |           |-- PatientRecords.csproj.FileListAbsolute.txt
|   |           |-- PatientRecords.csproj.GenerateResource.Cache
|   |           |-- PatientRecords.LoginPage.resources
|   |           |-- DesignTimeResolveAssemblyReferences.cache
|   |           `-- DesignTimeResolveAssemblyReferencesInput.cache
|   |-- Properties
|   |   |-- Resources.Designer.cs
|   |   `-- Resources.resx
|   |-- PatientRecords.csproj
|   |-- InsuranceInformation.cs
|   |-- InsuranceInformation.Designer.cs
|   |-- InsuranceInformation.resx
|   |-- LoginPage.cs
|   |-- LoginPage.Designer.cs
|   |-- LoginPage.resx
|   |-- MedicalHistory.cs
|   |-- MedicalHistory.Designer.cs
|   |-- MedicalHistory.resx
|   |-- PatientInformation.cs
|   |-- PatientInformation.Designer.cs
|   |-- PatientInformation.resx
|   |-- Search.cs
|   |-- Search.Designer.cs
|   `-- Search.resx
`-- PatientRecords.sln
```

Design Patterns

Validation.cs will contain all the code that validates inputs. Every form will access this class, and use it to validate input before sending anything to the database.

MyTools.cs will contain the code that creates the Database connection.

Commenting on all code so that someone can read the comments and grasp the idea of what the code was put there to do.

Database:
SQL Server 2014

Table Names:
Users- holds the user information of username and password
Patients - holds all the patient demographic information
Insurance- holds all the insurance information
MedicalHistory - holds all the medical history information

Relationships:
Users(1) --> Patients(Many)
Patients(1) -->Insurance(1)
Patients(1)-->MedicalHistory(1)

Coding Conventions

The naming structure for the C# Patient Records program will be camel case or pascal case depending on what is being named, examples shown below.
PatientRecords ← example of naming of Files.
Validation ← example of naming of Classes.
txtLoginUName ← example name of a text box input on the LoginPage which takes Usernames.
cmbWorkStatus ← example name of a combo box input on the PatientInformation page.

Variable names can be different depending on if they are public, or private variables.
fName ← would be the name of a private variable.
Fname ← would be the name of a public variable.

Database Column Names
Fname ← just like the public variable
MaritalStatus ← naming structure for two words
ColdSores ← on the form is shown as Cold Sores / Fever Blisters, in this situation we will just use the first selection of the two for its name.